**What Applicant is claiming**

The application has three independent claims, 1, 18, and 21, Claim 1 is addressed to "a method of adding a watermark to a sequence of executable instructions to render the sequence authenticable"; claim 18 and 21 are both addressed to "a method of authenticating a watermarked sequence of executable instructions" where the watermarked sequence of executable instructions has properties that result when the watermarking is done using the method of claim 1. Claim 1 can thus be regarded as exemplary for what Applicants are claiming. Claim 1 as presently amended reads as follows:

> **1.** (currently amended) A method of adding a watermark to a sequence of executable instructions to render the sequence authenticatable,
> the method comprising the steps of:
> receiving the sequence of executable instructions and a key; and
> using the key to modify the sequence of executable instructions so that the watermark may be obtained from the modified sequence, the sequence being modified such that the usefulness of the modified sequence for the sequence's intended purpose is not affected by the modifications made thereto and the watermark representing a watermark value ~~which may be employed to authenticate the sequence~~, alteration or absence of the watermark value being used when the sequence is authenticated to determine whether the sequence is authentic.

The first thing to be pointed out with regard to the claim is that the method "adds a *watermark* to a sequence of executable instructions to rend the sequence *authenticable*" (emphasis added). To understand Applicants' claims, it is necessary to understand these terms.

*Watermarking software*

As pointed out in Applicants' last response, the Collberg *Taxonomy* reference contains no mention whatever of watermarks. The Collberg *Software watermarking* reference defines watermarking in its *Abstract* as follows: "Watermarking embeds a secret message into a cover message". *Hiding* a message is thus the essence of watermarking. The use of the term *watermarking* in Applicants' Specification conforms to this

definition. See for example page 2, lines 8-10 of Applicants' Specification. Hiding may be done in many ways. In general, it is done by adding the message as noise to a digital representation of an audio signal, an image, or a video signal. The techniques enumerated on page 3 of the Office action of 1/21/05 insert watermarks in this fashion.

As pointed out at page 5, lines 15-18 of Applicants' Specification, techniques that add watermarks as noise cannot be used with code because code contains no noise. Instead, as described in Applicants' Specification beginning at page 5, line 19 and in Collberg, *Software watermarking,* there are two main techniques for software watermarking: *static* techniques, in which the code is rearranged to store the watermark, and *dynamic* techniques in which the code is rearranged so that the watermark is stored in execution state produced by an execution of the program.

*Authenticating software*

*Authentic* has many meanings in the dictionary. The one that comes closest to what is meant by the term in Applicants' Specification and indeed with regard to digital representations generally is this:

> conforming to an original so as to reproduce essential features (Webster's New Collegiate Dictionary, G. and C. Merriam Co., Springfield, MA, USA, 1977, p. 75)

Applicants' Specification speaks of authentication of software in terms that conform to the above definition:

> Digital representations are *authenticated* to make sure that they have not been altered in transit. Alteration can occur as a result of transmission errors that occur during the course of transmission from the source of the digital representation to its destination, as a result of errors that arise due to damage to the storage device being used to transport the digital representation, as a result of errors that arise in the course of writing the digital representation to the storage device or reading the digital representation from the storage device, or as a result of human intervention. (p. 10,. lines 1-5)

Thus, the method set forth in claim 1 adds a watermark to a sequence of executable instructions in such fashion that the watermark is hidden in the executable code or its

execution state and does so order that the watermark's value may be used to determine whether the sequence of executable instructions has been altered in transit.

*The body of the claim*

The body of the claim has a first step in which a key and a sequence of instructions are received and a second step in which the sequence of instructions is modified "so that the watermark may be obtained from the modified sequence". Because what is being modified is code, the code must be modified in the fashion set forth in the claim: "the sequence being modified such that the usefulness of the modified sequence for the sequence's intended purpose is not affected by the modifications". Further, what the watermark's message represents is "a watermark value which may be employed to authenticate the sequence." Finally, in the amended claim, "alteration or absence of the watermark value [is] used when the sequence is authenticated to determine whether the sequence is authentic". The claim's limitations thus absolutely require that the watermark is added to the code by "[modifying] the sequence of instructions", that the watermark represents a value which is used to authenticate the sequence of code, and that alteration or absence of the watermark value is used to determine whether the sequence is authentic.

**The rejection of claim 1 as anticipated by Collberg *Taxonomy***

The reference that Examiner applies in his rejection of claim 1 under 35 U.S.C. 102 is the Collberg *Taxonomy* reference. All that this reference has in common with claim 1 is that adding a watermark to code and obfuscating code both result in modification of the code. The modifications are, however, for different purposes and done in different manners. Obfuscation as described in Collberg *Taxonomy* is done to make reverse engineering of the code harder (Collberg *Taxonomy* page 3, second column, paragraphs 3 and 4), while Applicants' modifications are done to watermark the code and thereby make it authenticatable. There is no indication whatever in Collberg *Taxonomy* that obfuscation as described there has anything whatever to do with watermarks or with authentication. Collberg's obfuscator does not take a key as an argument, and it is therefore difficult to see either how it can add a watermark to the code or how any watermark it did add could

be used for authentication. Collberg's FIG. 6, cited by Examiner for the proposition that Collberg uses a key, takes as arguments the code being obfuscated and indications of the cost and potency of the desired obfuscation. Because Collberg *Taxonomy* discloses nothing whatever about watermarking a sequence of executable instructions, using a key to do so, or using the watermark's message to authenticate the sequence of executable instructions, the reference cannot anticipate Applicants' independent claims 1, 18, and 21.

**Claim 1 and the Collberg *Software Watermarking* reference**

*The disclosure of Collberg Software Watermarking*

As the title indicates, Collberg *Software Watermarking* does disclose watermarking software and the use of keys to do so. The purpose of the watermarking is to embed a copyright notice or customer identification number in the software (Introduction, first paragraph). There is simply no notion in Collberg, *Software Watermarking* that a watermark might be used to *authenticate* the software. The character string "authentic" does not appear in the reference. The closest that Collberg, *Software watermarking* comes to the topic of authentication is the discussions of tamperproofing watermarks in sections 2.3 and 5.5 and 5.5.1. Tamperproofing a watermark is taking measures to make sure that the watermark cannot be removed from the code and this, of course, is not the same as tamperproofing the code that carries the watermark. Section 5.5.1 does include one example which shows Java's class reflection mechanism may be used to detect tampering with a watermark based on a graph's node type and the program may be made to terminate on detection.

What Collberg, *Software watermarking* is chiefly concerned with is the difficulty of watermarking something which is as malleable as executable code. For example, static watermarks in executable code can be removed simply by obfuscating the executable code until the watermark is no longer detectable (see Section 2.3). For that reason, Collberg, *Software watermarking* prefers dynamic watermarking, but even there, obfuscation can render most dynamic watermarks undetectable (see Section 5.) In

Section 5, Collberg, *Software watermarking* discloses a new watermarking technique called "dynamic graph watermarking" which has more resistance to obfuscation attacks. The conclusion which must be drawn from Collberg, *Software watermarking* is that only the most complex dynamic watermarking techniques are of any use in protecting executable code.

*Watermarking and authentication*

While Collberg's pessimism about watermarking executable code may be justified when the watermark is used to show ownership, Applicants have demonstrated that even simple static watermarking is an effective way to *authenticate* executable code. The reason that this is so is that in authentication, loss or corruption of the watermark is *proof that the code has changed since it was watermarked.* Thus, the very property of watermarked code that renders the watermark almost useless for showing ownership of the code makes the watermark extremely useful for detecting faulty transmission of the code or tampering with the code and therefore for authenticating the code. Claim 1 as amended clearly sets forth that authenticating code according to Applicants' method involves checking the watermark for loss or corruption.

*Patentability of claim 1 over Collberg Software Watermarking*

Applicants' claim 1 clearly sets forth in its preamble that the watermark is added to the sequence of executable instructions "*to render the sequence authenticable*" and sets forth in the body of the claim that "the watermark represent[s] a watermark value which may be employed to *authenticate the sequence*". Finally, the technique set forth in claim 1 for using the watermark value for authentication, "alteration or absence of the watermark value being used when the sequence is authenticated to determine whether the sequence is authentic", has no equivalent whatever in Collberg, where watermarks are only used to show ownership. Since Collberg *Software Watermarking* discloses nothing whatever about authentication using watermarks or about Applicants' technique for using the watermark value for authentication, the reference does not disclose all of the limitations of the claim and cannot anticipate the claim. Further, because neither Collberg

*Taxonomy* nor Collberg *Software Watermarking* discloses anything whatever about authentication, the combined references do no disclose all of the limitations of claim 1.

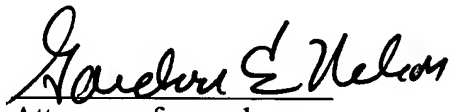
**The IDS**

Applicants have been unable to locate complete versions of two of the references cited in the PCT application that was the parent of the present application. The *Digimarc Watermarking Guide* is currently available on the Digimarc Web site in a version that is copyrighted 2004, and that version is included in the IDS. The copy of the Zhao article, "Look, It's not There", *Byte Magazine*, January, 1997 was obtained from the magazine's archives, but it is fragmentary.

**Conclusion**

Applicants have filed the present RCE with the required fee and a submission that amends the claims to better distinguish them from the references cited in the final rejection of 1/21/05 and shows why the claims as amended are patentable over those references. Applicants have further included a *Petition for a 1-month extension of time* with the fee required for the extension. Applicants have fulfilled the conditions of 37 C.F.R. 1.114 and respectfully request that Examiner withdraw the finality of his rejection and continue the examination of this application.

Respectfully submitted,



Attorney of record,
Gordon E. Nelson
57 Central St., P.O. Box 782
Rowley, MA, 01969,
Registration number 30,093
Voice: (978) 948-7632
Fax: (866)-723-0359
5/23/05
Date

## Certificate of Mailing

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to:

Commissioner for Patents

P.O. Box 1450
Alexandria, VA 22313-1450

on 5/23/05
(Date)
Gordon E. Nelson, #30,093
Gordon E. Nelson
(Signature)